USER TASK INTERFACE IN A WEB APPLICATION

RELATED INVENTIONS

The present invention is related to the inventions disclosed in the following commonly-assigned U.S. patent applications Ser. No. 10/795,008 (now U.S. Pat. No. 7,493,563), entitled "Using Content Aggregation to Build Administration Consoles", which was filed on Mar. 05, 2004; Ser. No. 10/795,007 (now U.S. Pat. No. 7,444,633), entitled "Federating Legacy/Remote Content into a Central Network Console", which was filed on Mar. 05, 2004; and Ser. No. 10/754,375, entitled "Dynamic Composition of Help Information for an Aggregation of Applications", which was filed on Jan. 9, 2004.

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to computing systems, and deals more particularly with techniques for providing a task interface for users of Web-accessible applications, such as console applications.

2. Description of the Related Art

Computer software and hardware systems are often configured, monitored, and managed by one or more administrators using graphical user interfaces called "consoles". Often, each system component within an information technology ("IT") environment has its own independently-developed console for carrying out these operations. Even a relatively small business can require a number of different computer-based products (including hardware components and/or software components) for its business solution, and a large business or other enterprise may have a very large number of such products in its IT environment.

A number of problems with prior art consoles have been described in the related inventions titled "Using Content Aggregation to Build Administration Consoles" (U.S. Pat. No. 7,493,563) and "Federating Legacy/Remote Content into a Central Network Console" (U.S. Pat. No. 7,444,633), and solutions to those problems are described therein.

A problem not addressed in the related inventions is that Web-based consoles have been limited in their capabilities because of the lack of a standard, shareable infrastructure to support task-based user interfaces. Task-based user interfaces are quite common today in desktop computing environments and operating systems based on a windowing paradigm where a window is created to represent each active user task. Applications could build their own task-based mechanism, but that is analogous to having to build both the application user interface and the desktop user interface within the application. This build-your-own approach also does not facilitate interactions among application user interfaces or the ability to easily switch from a task in one user interface to a task in another user interface without loss of data, both of which are features that users expect to be present in desktop computing environments.

Without a task-based user interface for Web-based consoles, it may be difficult for users of those consoles to achieve optimum productivity levels. The need for task-based Web user interfaces is not restricted to administration consoles, however. For example, some applications such as word processors may be presented as Web applications. As in a desktop computing environment, a user can be productive if he can manage different instances of the word processor application for different documents that he switches between. Accordingly, what is needed are improvements that provide task-based user interface features in Web-based applications.

SUMMARY OF THE INVENTION

An object of the present invention is to provide task-based user interface features in Web based applications.

Another object of the present invention is to provide techniques that enable users of Web-based user interfaces to have multiple instances of a particular activity in progress at any given time, with independent state information maintained for each instance.

A further object of the present invention is to enable users of Web-based applications to be interrupted from a first activity to switch to a second activity, without losing state when later returning to the first activity.

Still another object of the present invention is to provide techniques that enable activities carried out in a Web-based application to have a life cycle with a definite beginning and end.

Yet another object of the present invention is to provide users with a capability for persisting custom tasks comprised of dynamically-built aggregated content and contextual information.

A further object of the present invention is to enable users to launch a task from a task-based user interface and programmatically pass contextual information to the launched task.

Other objects and advantages of the present invention will be set forth in part in the description and in the drawings which follow and, in part, will be obvious from the description or may be learned by practice of the invention.

To achieve the foregoing objects, and in accordance with the purpose of the invention as broadly described herein, the present invention may be deployed as methods, systems, and/or computer program products embodied on one or more computer-readable media. In one aspect, the present invention provides a task-based interface for a Web-based application by creating a new instance of a work unit for which a rendered view is requested, where the view comprises content renderable by one or more content-creating software entities; and associating state information with the new instance, thereby enabling one or more additional instances of the work unit to be created, each maintaining its own independent version of the associated state information. Creating the new instance and associating the state information may be repeated for one or more distinct work units.

An entry representing each of the instances is preferably rendered in a user interface control, such that any of the instances can be viewed by selecting its rendered entry. The user interface presents a view of a particular work unit, and the view of the particular work unit is preferably overlaid with a view rendered by the newly-created instance.

Content may be added dynamically to the new instance, wherein the dynamically-added content is created by a dynamically-selected content-creating software entity.

With the creation of a work unit or dynamically-added content, contextual information may be passed as well so that the content may be rendered in context of the current activity. After the initial presentation of content, the contextual information may be refreshed to permit the rendering in a new context.

A user may close a view associated with any of the distinct work units or the new instance or additional instances thereof, while retaining the associated state information for the work units not being closed. In addition or instead, the user may remove the dynamically-added content from the new instance.